Mikael Bosund

Responsiiviset web-tekniikat

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

12.5.2014

Metropolia

Tiivistelmä

Tekijä Otsikko Sivumäärä Aika	Mikael Bosund Responsiiviset web-tekniikat 40 sivua 12.5.2014
Tutkinto	insinööri (AMK)
Koulutusohjelma	mediatekniikka
Suuntautumisvaihtoehto	digitaalinen media
Ohjaajat	lehtori Ilkka Kylmäniemi yliopettaja Pentti Viluksela

Insinöörityön tavoitteena oli kartoittaa responsiivisten web-tekniikoiden kehityshistoriaa ja moderneja tapoja tuottaa responsiivisia sivuja parhaita nykykäytäntöjä hyödyntäen. Web-tekniikat lähtivät ulkoasultaan staattisia pikselipohjaisista kokomäärityksistä ja toisto-laitteiston kehitys oli oleellisessa asemassa web-tekniikan kehityksessä.

Työssä perehdyttiin responsiivisuuden kannalta oleellisimpiin tekniikoihin ja siihen, kuinka erilaiset mediaformaatit vaativat erillistä huomiota käyttöönotossa. Tutkittaviksi ydintekniikoiksi valittiin HTML5- ja CSS3-kielet, jotka edustivat kaiken web-sivustotuotannon senaikaisia nykystandardeja. Nämä loivat parhaat käytännöt tuottaa eri selainten ja laitteiden kanssa yhteensopivia responsiivisia sivuja.

Insinöörityössä tutkittiin kolmea erillistä sivustoa, jotka kaikki hyödyntävät responsiivista suunnittelua mutta joiden sisältö ja käyttötarkoitus poikkesivat toisistaan huomattavasti. Projekteista kaksi tehtiin Joomla-julkaisujärjestelmään ja yksi ohjelmoitiin kokonaisuudessaan modulaarisista osasista, joista sivut koottiin PHP-tekniikan avulla. Projektia varten tuotetut sivustot on kaikki julkistettu, kaksi niistä kansainväliseen käyttöön. Ne saivat asiakkailta paljon myönteistä palautetta, minkä vuoksi tekniikoiden valintaa voitiin pitää onnistuneena.

Työn tuloksena voidaan nähdä, kuinka responsiiviset tekniikat eivät ole enää vain yksi erityistapaus suunnitella sivustorakennetta vaan enemmänkin tulevaisuuden tapa, jolla pystytään vapautumaan laite- ja selainyhteensopivuuteen liittyvistä ongelmista.

Avainsanat	responsiivinen web, HTML5, CSS3, web-media

Abstract

Author Title Number of Pages Date	Mikael Bosund Responsive Web-technologies 40 pages May 12th 2014
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital media
Instructors	Ilkka Kylmäniemi, Lecturer Pentti Viluksela, Principal Lecturer

The goal of the thesis was to study the history of development of Web-technologies and modern means to produce responsive web-sites using best modern practices. Web-technologies started to develop from static layouts designed in pixel formats, and the development of display equipment played a key role in the evolution of Web technologies.

The thesis studied the most fundamental technologies of modern Web development and challenges of deploying various media formats. The chosen core technologies were HTML5 and CSS3 languages, which were the modern standards for all Web development of the time. They enabled the best means to produce responsive sites compatible with the most common browsers and devices.

To gain broad perspective of the topic, three different responsive Web-sites were researched. Each of these sites had greatly varying content and purpose of use. Two sites were built on Joomla content management system and the third was programmed from modular files that were combined into the final page using PHP technology. All sites produced for the project are in published use, two for international users. They received much positive feedback from the customers, which indicates that the chosen technologies were successful.

The conclusion of the thesis is that responsive technologies are no longer just an exceptional case of site stuctural design. Instead they should be considered the new standard of the future which can be used to avoid device and browser compatibility issues.

Keywords	Responsive Web, HTML5, CSS3, Web-media

Sisällys

1 Johdanto 1

2 Kohti responsiivista webiä 2

2.1 Näyttötekniikoiden historia 2
2.2 Web-tekniikoiden historia 3
2.3 HTML5 ja tekniikan murros 4

3 Responsiiviset tekniikat 5

3.1 Suunnittelun rakennemallit 5
3.2 Viewport-asetus 6
3.3 Kokomääritykset 7
3.4 Media queryt ja breakpointit 9
3.5 Pikselikuvat 10
3.6 Vektorikuvat 11
3.7 Responsiivinen web mobiililaitteissa 13

4 Insinöörityön tutkimuskohteet 14

4.1 (Salainen osio)
4.2 (Salainen osio)
4.3 Netrakin kotisivu (modulaarinen PHP) 29

5 Yhteenveto 37

Lähteet 38

1 Johdanto

Web-sivujen kehitystavat ovat siirtyneet teknisesti suureen siirtymävaiheeseen, jossa vanha staattinen web-teknologia on syrjäytymässä uuden responsiivisen eli mukautuvan web-suunnittelun tieltä. Yrityksille ja muille webissä toimiville tahoille on hyvin tärkeää, että niiden palveluita pystytään käyttämään laiteriippumattomasti ja mobiilisti, myös poissa toimistopäätteeltä. [1; 2; 3.]

Samaan aikaan nopeasti kasvava mobiililaitteiden kirjo teknisine kykyineen ja rajoituksineen luo asiaan perehtymättömälle web-kehittäjälle huomattavan haasteen saada sivut toimimaan laiteriippumattomasti. Koska täydellinen toistuvuus kaikissa laitteisto- ja sovellusympäristöissä ei ole enää useimpien sivustoprojektien realistinen tavoite, on web-kehitys ohjautunut kohti mallia, jossa palvelun tekninen toistuvuus pyritään mahdollistamaan lähtökohtaisesti mahdollisimman laajalle yleisölle. Tähän parhaat eväät tarjoaa nykyään HTML5-tekniikka, joka on valittu nykyiseksi alan standardiksi web-sivujen kehitystyössä ja jolla on nykytekniikoista laajin selaintuki. [1; 2; 3.]

Insinöörityön tavoitteena on perehtyä moderneihin responsiivisen web-suunnittelun standarditekniikoihin ja tapoihin, joiden avulla sivustoja voidaan tuottaa oikeaoppisesti, jotta toimivuus voidaan taata suurimassa osassa laitteistoa. Työ alkaa tekniikoiden historiaosiolla, jossa pyritään tutkimaan, miten erilaiset tekniikat ovat kilpailleet standardiasemasta heikkouksineen ja vahvuuksineen sekä mitkä niistä ovat selviytyneet nykypäivään saakka. Toisessa osassa käydään läpi keskeisiä nykykäytäntöjä, joilla HTML5-standarditekniikan ominaisuuksia hyödynnetään responsiviisessa suunnittelussa. Tärkeänä seikkana pyritään kartoittamaan, mitä responsiivinen suunnittelu oikeastaan tarkoittaa teoriassa ja toistolaitteiden kannalta, sekä luodaan katselmus tekniikoiden tulevaisuuteen.

Viimeinen osio kattaa projekteja, joissa käydään läpi, miten responsiivisia sivuja suunnitellaan yksittäiselle sivustolle, julkaisujärjestelmää hyödyntävälle massiivisemmalle yrityssivustolle sekä kymmeniä sivupohjia kattavalle järjestelmälle, joka on rakenteellisesti optimoitu nopeaan sisällöntuotantoon. Kaikissa näissä projekteissa työskentelymetodit poikkeavat toisistaan huomattavasti, mutta niiden yhdistävä tekijä on responsiivisuus.

2 Kohti responsiivista webiä

2.1 Näyttötekniikoiden historia

Ensimmäisten HTML-versioiden aikaan 1990-luvun taitteessa suurin osa tietokoneista käytti CRT- eli kuvaputkinäyttötekniikkaa. Näyttöjen ominaispiirteitä olivat suuri koko, rajallinen terävyys, ruudunpäivitys ja 4:3-kuvasuhde. Näyttöjen tukemat resoluutiostandardit olivat VGA (640 x 480), SVGA (800 x 600) ja XGA (1024 x 768) sekä yleiset näyttökoot 15—19” välillä. CRT-näyttöjen aikaan digitaaliseksi kuvakoon standardiksi määritettiin 72 ppi:n tarkkuus, jolla ei kuitenkaan ollut käytännön merkitystä, koska digitaalinen grafiikka laskettiin tavallisesti pikseli- eikä fyysisessä koossa. [4; 5.]

LCD-näyttöjen alkaessa yleistyä ei suurempien näyttöjen valmistus ollut enää niin kallista ja tilaa vievää, jolloin näyttöjen tukema resoluutio alkoi pikkuhiljaa kasvaa. Yleisimmäksi kuvasuhteeksi vakiintui 16:9, ja resoluutio vaihteli usein HD 720 -standardista (1280 x 720) HD 1080 -standardiin (1920 x 1080). LCD-näyttöjen pikselitiheys oli noin 100 ppi:n tarkkuudella, eivätkä ne poikenneet CRT-näytöistä kovin merkittävästi, joten tekniikalla ei ollut suurta vaikutusta digitaaliseen grafiikantuotantoon tai web-suunnitteluun. [4; 6.]

2000-luvulla monikäyttöisten älypuhelinten tulo markkinoille loi valmistajille painetta kehittää tarkempia näyttöjä ja näyttötekniikoiden kehityksen suunta alkoi siirtyä pois PC-tietokoneiden pöytänäytöistä. Ensimmäisen sukupolven älypuhelimet olivat resoluutioltaan pienikokoisia, mutta niiden pikselitiheys kykeni ylittämään jo pöytä-näytöt. Vuonna 2007 Apple julkaisi iPhone 2G -puhelimensa, joka muodostui hetkelliseksi mobiilistandardiksi. Puhelimessa oli 3,5 tuuman näyttö 320 x 480 pikselin resoluutiolla ja 163 ppi:n pikselitiheydellä, eli noin 1,5-kertaisella tarkkuudella pöytänäyttöihin nähden. [7; 8.]

2010-luvulla alalla vallitseva kova kilpailu on moninkertaistanut uusien laitteiden näyttötarkkuudet nopeasti, mutta koska mobiililaitteen koko on rajattu, se merkitsee pikselien koon pienentymistä ja käänteisesti pikselitiheyden kasvua. Vuoteen 2014 mennessä mobiililaitteiden resoluutiostandardeja oli paljon laitteiden valmistajien suuren määrän vuoksi, mutta näyttöjen tarkkuus liikkui tavallisesti 300:n ja 450 ppi:n välimaastossa eli 3—4 kertaa pöytänäyttöjä tarkempana. [7.]

2.2 Web-tekniikoiden historia

Web-sivustojen alkaessa yleistyä 1990-luvun puolivälissä oli käytössä HTML2.0-tekniikka, jolla pystyi suunnittelemaan ja tekemään yksinkertaisia HTML-sivutiedostoja. Näihin tiedostoihin pystyi upottamaan tekstiä, kuvia ja yksinkertaisia määrityksiä, joilla sivustojen värejä, kokoja ja muita perusparametrejä pystyi vaihtamaan. Keväällä 1998 tekniikkaa päivitettiin HTML4.0.1-versioon, joka laajensi tekniikkaa tukemaan suurempaa kirjoa visuaalisia elementtejä, jolloin ulkoasusuunnittelun merkitys kasvoi. [9; 10.]

2000-luvun alussa sivut suunniteltiin usein Photoshopissa tai muissa rasterigrafiikkaa hyödyntävissä ohjelmissa, minkä jälkeen elementit luotiin HTML-koodiin pikseliformaatissa. Tällöin sivun ulkoasu pysyi lukittuna kuvaruudun pikseleihin, jolloin rakenne-elementit, teksti ja kuvat pysyivät hyvin koossa toisiinsa nähden. Tämä toimintatapa teki kuitenkin sivuista joustamattomia, mutta koska silloiset näyttötekniikat noudattivat pääasiassa muutamaa kokostandardia, ei tekniikan laajentamiseen ollut tarvetta. Hyväksi suunnittelutavaksi web-suunnittelijoiden keskuudessa kehittyi tapa lukita sivujen leveys senaikaisten pienimpien näyttökokojen mukaan (esimerkiksi 800 pikselin leveyteen) ja antaa sivun sisällön liikkua pystysuunnassa, kuin luettaessa pitkää paperiarkkia. [10; 11; 12; 13.]

HTML4-tekniikkaa tukevien mobiililaitteiden alkaessa yleistyä syntyi web-suunnittelun ensimmäinen murrosvaihe, jossa mobiililaitteita kohdentaville sivuille luotiin vaihtoehtoinen mobiiliversio. Tämä oli yleensä kokonaan erillinen sivusto, josta kookkaammat sisältöelementit oli karsittu pois, jotta sivun pystyi lataamaan nopeasti hitaillakin mobiiliyhteyksillä. Sivut siis sisälsivät mahdollisimman vähän kuvia, videoita tai muita graafisia elementtejä tai ulkoisia toiminnallisuuksia. Monien mobiilisivustojen valikkorakenteet rakennettiin prosentuaaliseen kokoon, jolloin sivu skaalautui suoraan näytön leveyteen. Tämä rytmitti tekstin ja muun sisällön automaattisesti sen mukaan, jolloin pystyttiin näyttämään kapeillakin näytöillä eikä sivuttaisuuntaisia vierityspalkkeja ilmestynyt sivuille. [11.]

Käyttäjien näyttöjen tarkkuudet olivat kuitenkin kasvussa sekä mobiililaitteiden että perinteisten pöytänäyttöjen alueilla, mikä loi kasvavaa ongelmaa niin mobiili- kuin staattisissa pääsivustoissa. Alkoi olla selvää, ettei HTML4 enää kyennyt tekniikkana vastaamaan moneen suuntaan kasvaville vaatimuksille, ja tämä alkoi luoda kasvavaa tarvetta korvata vanhentunut työympäristö uudella standardilla. [1; 2; 3.]

2.3 HTML5 ja tekniikan murros

Kun HTML4:lle etsittiin uutta jatkajaa, kilpaili johtoasemasta muutama eri tekniikka:

Alun perin Macromedian erillisenä multimediatyökaluna kehittämän **Flash-tekniikan** etu oli, että se tuki kaikkia multimedian muotoja, minkä ansiosta sille oli muodostunut vankka asema esimerkiksi web-videoiden soitintyökaluna. Ongelmana Flashissa oli kuitenkin sen sisäinen rakenne; Flash-tiedostot käyttivät hyväkseen erillistä ohjelmointi- ja julkaisuformaattia, joka esti muun muassa hakukonerobottien pääsyn tutkimaan Flash-elementin metatietoja. Flashin asemaa pahensi myös se, ettei se Adoben alaisena yrityksenä toiminut vapaan koodin periaatteiden mukaisesti vaan hyödynsi omia ohjelmointikieliään. Pian sen jälkeen, kun Apple esti mobiililaitteissaan tuen Flash-sovelluksille, ilmoitti Adobe lopettavansa Flashin päivityksen mobiililaitteissa. Tätä voidaan pitää hetkenä, jolloin Flash menetetti sijansa tulevaisuuden web-tekniikkana, jonka avainsana on laiteriippumattomuus. [14.]

Syksyllä 2007 Microsoft julkaisi Flashin kanssa kilpailevan tekniikan **Silverlightin**. Aluksi Silverlightia käytettiin lähinnä videosoittimena suoratoiston (streaming media) tarkoituksissa, esimerkiksi Netflix-palvelussa, josta se laajeni tukemaan mediaformaatteja laaja-alaisesti. Silverlight ilmestyi mukaan sen verran myöhäisessä vaiheessa, ettei markkinoilla ollut enää suurta tarvetta uudelle standardille ja tekniikan käyttöönotto jäi vähäiseksi. Osittain käytön harvinaisuuden vuoksi, osittain ehkä huonon julkaisuajankohdan vuoksi monet älypuhelinvalmistajat jättivät puhelimistaan pois Silverlight-tuen, minkä vuoksi juuri mullistuksessa olevassa web-tekniikassa turvauduttiin entistä varovaisemmin vaillinaista selaintukea hyödyntäviin tekniikoihin. [15; 16.]

HTML(2—4), CSS(1—2) ja Javascript-tekniikat olivat olleet web-toteutuksen peruspylväät jo lähes 20 vuotta, mutta jatkuvat selaintukiongelmat ja vanhentunut toimintamalli tarkoittivat, että web-kehittäjien oli pakko hyödyntää vaihtoehtoisia lisätekniikoita tiettyjen toiminnallisuuksien luomiseksi. Vastauksena tekniikan heikkouksiin WHATWG alkoi kehittää HTML4.0:n pohjalta uutta **HTML5-määritelmää**, jonka tarkoitus (hyödyntäen cascading stylesheet 3:n, eli CSS3:n määrityksiä) oli standardisoida repaleista HTML4-ohjelmointikieltä. Lisäksi se sisälsi erilaisia toiminnallisia ominaisuuksia, joihin tarvittiin aiemmin Javascript-kieltä. Tekniikkaa oli mietitty myös responsiivisuuden sekä laiteriippumattomuuden kannalta, ja se sisälsi jo julkaisuvaiheessaan suuren määrän määrityksiä, joilla responsiivisten sivujen tuotanto yksinkertaistui. [1; 2; 17.]

3 Responsiiviset tekniikat

Yksi responsiivisen suunnittelun peruspilareista on täysi *laiteriippumattomuus*, joka tarkoittaa, että yhtä sivua voidaan käyttää kaikissa toistolaitteissa sen resoluutiosta tai pikselitiheydestä riippumatta. Tämä ei tarkoita, että sivut toistuisivat kaikkialla samanlaisena ja esimerkiksi samalla palstajaolla, vaan sillä pyritään määrittämään tapa, jolla sivurunko pystyy mukautumaan erilaisiin leveyksiin huomioiden käyttäjän kannalta tärkeät parametrit, kuten tekstin luettavuuden ja ulkoasun harmonian säilymisen. [1; 3; 18.]

Koska HTML5:n ohella mikään muu web-tekniikka ei tarjoa kunnollisia työkaluja responsiiviselle suunnittelulle, on toteutus tehtävä HTML5-tekniikan avulla. HTML5 ei kuitenkaan itsessään takaa, että sivut toimisivat responsiivisesti, vaan web-kehittäjän on tunnettava tekniikat. Niitä oikein hyödyntämällä ja suunnittelemalla rakenteen oikeaoppisesti on sivujen responsiivisuus toteutettavissa. [1; 3; 18.]

3.1 Sunnittelun rakennemallit

Responsiivisia sivuja suunniteltaessa tärkeää on, että sivut kehitetään HTML5-standardin mukaisilla tavoilla. Sivurakenteessa pitäisi hyödyntää mahdollisimman laaja-alaisesti HTML5:n elementtijakoa sisällön, navigaation ja muiden perusosien jaottelussa, eli suosia standardin mukaisia rakenne-elementtejä ja toteutusratkaisuja. [1; 2.]

Yksi suunnittelua vaikeuttava perusasia on, että sivuston ohjelmointikoodi on yksiulotteinen ja visuaalinen puoli kaksiulotteinen. Tämän vuoksi voi joskus sivurakenne olla vaikea hahmottaa, ja varsinkin uusia responsiivisia ulkoasuja suunniteltaessa rakennekaavioiden piirtäminen auttaa työtä huomattavasti. Rakennemallista sisältöelementit kannattaa jakaa niiden keskinäisiä kokosuhteita kuvaaviin laatikoihin sekä selvittää, mitkä osiot vaativat erillisiä rakenne-elementtejä ympärilleen. Rakennemallista tulisi selvitä, mitkä elementit sivuilta löytyvät ja miten ne asettuvat sivuille, kun näkymä vaihtuu merkittävästi esimerkiksi työpöytäkoosta mobiilikokoon. [2.]

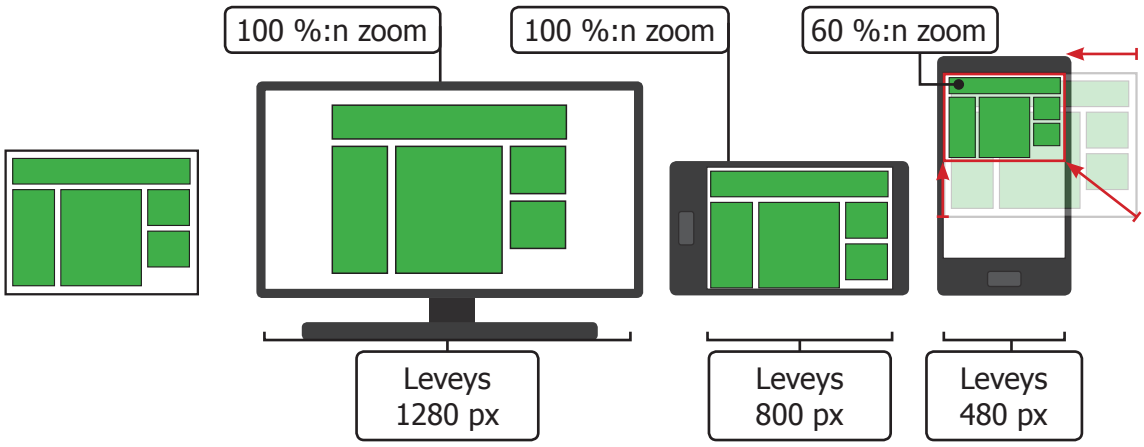
Rakennemalleja voi tehdä nykyään monilla ohjelmilla, tai käsin (yleensä ruutupaperille) piirretty malli voi toimia hyvänä havannointikuvana. Photoshop- tai muuta pikseliohjelmaa käytettäessä kannattaa muistaa, ettei responsiivisesta sivusta kannata tehdä pikselilleen täydellistä esitystä, sillä sitä on harvoin realistista toteuttaa. [2.]

3.2 Viewport-asetus

Viewport on yksi responsiivisen suunnittelun ydinasetuksista. Sillä kuvataan selainikkunan kokoa 100 % zoom-tasossa. Tavallisessa PC-tietokoneen web-selaimessa zoom-tasoa muutetaan harvemmin, joten viewport on lähes aina suoraan selainikkunan kokoinen. [2.]

Nykyisissä mobiililaitteiden web-selaimissa viewport on suunniteltu hyödyntämään erilaista zoom-tasoja, joilla kuvaruudut kykenevät toistamaan tavalliset sivukoot asettamalla ne ensin näyttökokoa suurempaan viewport-ikkunaan ja skaalaamalla näkyvää sivua pienemmäksi, jolloin kokonaisuuden voi nähdä helpommin. Viewportien tarkka koko vaihtelee jokaisen selaimen mukaan, mutta yleisesti se on oletusarvoisesti suurempi kuin mobiililaitteen näytön maksimiresoluutio. [2; 19.]

Kuva 1 esittää, miten viewport-asetus mukauttaa sivun erilaisissa toistolaitteissa. Suurimmissa näytöissä sivu yleensä keskitetään selainikkunaan, ja kun leveys ei riitä toistamaan sivua kokonaan, koko sivusto skaalautuu pienempään kokoon.



Kuva 1: Viewportin toimintalogiikka [2].

Viewport on tärkeä responsiivisessa suunnittelussa, koska sen oletusarvoja voidaan muuttaa, jolloin pystytään tarkemmin määrittämään sivuston käyttäytyminen erilaisissa laitteissa. Viewportille on luotu kaksi määritystapaa:

1. **Viewport meta tag**, joka asetetaan head-elementin sisään. Meta tagissa viewportin määritykset upotetaan attribuuttiin ”content”, joka voi sisältää useita eri määrityksiä. Meta tagin heikkona puolena on, ettei siihen voida laittaa määrityksiä relatiivisille yksiköille, koska kaikki attribuuttien numeroarvot käsitellään vanhahtavasti pikseliarvoina ja ilman yksikkömerkintää. Vahvana puolena meta tag -pohjainen viewport-määritys on selainversioissa erittäin laajalti tuettu standardi ja sen toiminta on yleensä varmempaa kuin CSS-määrityksien (tammikuu 2014). [2; 19.]

2. **@viewport CSS-määritys**, joka löytyy jostain kohtaa CSS-tyylitiedostoa, yleensä ennen media query -määrityksiä. CSS-tekniikka tukee kaikkia yksiköitä ja määrityksiä, mutta sen selaintuki on vielä vaihtelevaa. Tulevaisuudessa web-standardin mukaisen CSS-määrityksen voidaan olettaa korvaavan viewport-metatagin, mutta mikäli viewport-määritys ja varsinkin mobiiliympäristön toistuvuus on tärkeä, on toistaiseksi varmempaa käyttää viewport-metatagia (tammikuu 2014). [2; 19.]

3.3 Kokomääritykset

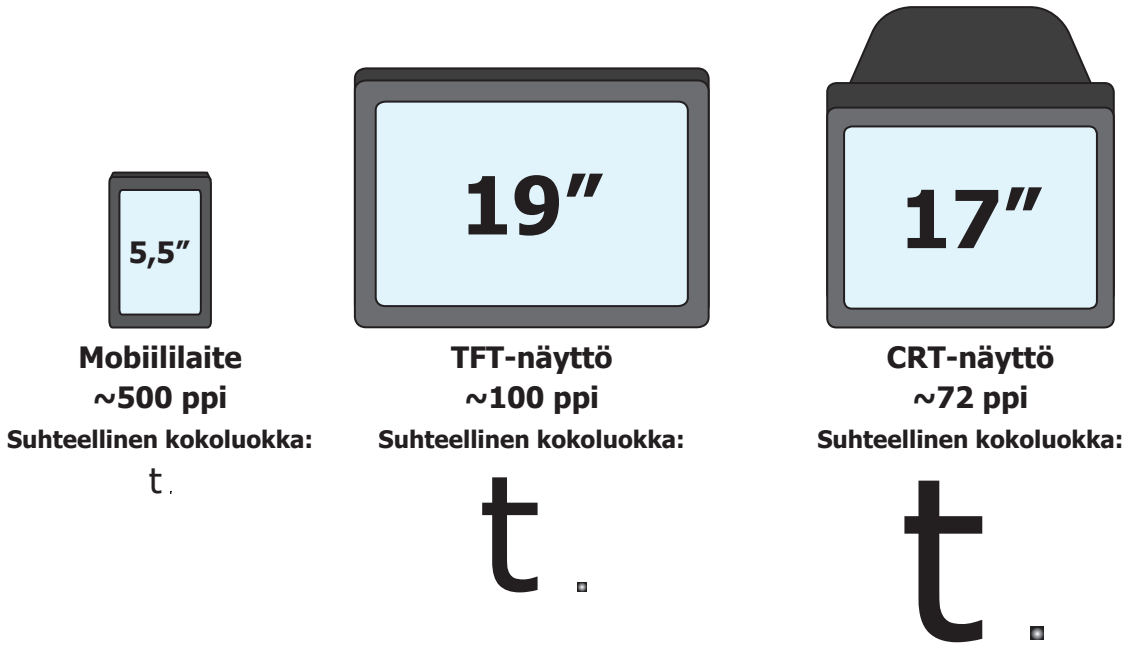
Perinteinen **pikselipohjainen** kokomäärittely pohjautuu nimensä mukaisesti siihen määrään pikseleitä, jonka elementti vie kuvaruudulta. Pikselin koko ei kuitenkaan ole vakio, joten sen absoluuttinen koko vaihtelee hyvinkin paljon. 1980-luvulla määritelty ensimmäiseen Macintosh-tietokoneeseen pohjautuva standardi oli 72 **ppi** (pixels per inch tai pikseliä per tuuma). [5.]

Pikselin koon voi laskea seuraavalla laskukaavalla:

$$\frac{2{,}54\ {}^{cm}/_{in}}{72\ {}^{px}/_{in}} = 0{,}035\ {}^{cm}/_{px} \approx 0{,}4\ {}^{mm}/_{px}$$

Yleisesti leipätekstissä käytetty fonttikoko 12 px tarkoittaa, että fontin maksimikorkeus kuvaruudulla on 12 pikseliä, joka määrittää samalla suhteessa kirjainten leveyden. 72 ppi:n tarkkuutta hyödyntävällä näytöllä saadaan merkkien maksimikorkeudeksi laskettua 4,2 mm, jonka lukeminen on hyvin vaikeaa. Tämän hetken (tammikuu 2014) markkinoiden huippusaavutus mobiililaitteelle on 538 ppi:n tarkkuus, jonka pikseli on lähes 7,5 kertaa perinteistä digitaalista standardia pienempi (n. 0,05 mm:n korkuinen). Tällöin saman 12 px:n kokoisen fontin korkeus on 100 %:n koossa alle 0,6 mm, joka on lukukelvoton ja vaatii käyttäjältä huomattavaa zoom-tason muutosta, jotta sivuston tekstejä pystyisi lukemaan vaivatta. [2; 5; 20; 21]

Kuvassa 2 esitetään 16 px:n pikselikokoon määritetyn t-kirjaimen ja yhden pikselin suhteellista kokoa erilaisissa näytöissä.



Kuva 2: Kirjainten ja pikselien koot mittakaavassa [2].

Koska pikselien fyysinen koko vaihtelee huomattavasti, on nykyisen laiteriippumattoman responsiivisen suunnittelun kulmakiveksi muodostunut **relatiivinen kokomääritys** 'em'. Em-pohjaisessa määrittelyssä toistolaite luo fontille tarkan koon typografisen 'pt'-skaalan mukaan, jossa 12 pt = 1 em = 16 px (pikselimääritys perinteisissä, n. 100 ppi:n näytöissä). Koska em huomioi kuvaruudun pikselitiheyden, sen lopputulos on lähes aina sama riippumatta toistolaitteesta. [1; 2; 21.]

Toisin kuin pikseliden kanssa, laskettaessa em-kokoja on muistettava, että toistolaitteet osaavat käsitellä em-pohjaisia desimaalilukuja äärettömän tarkasti ja toistaa ne kaikilla selaimilla ja kuvaruuduilla vaivatta. Kaikki nykyhetken (tammikuu 2014) näytöt hyödyntävät kuitenkin kuvapisteitä laskiessaan lopullista toistokokoa. Täten vaikka em onkin luku-arvoltaan suurempi kuin pikseli, on turhan monen desimaalin (>3 desimaalia) merkintä lähes hyödytöntä, kun desimaalit edustavat pikselin murto-osia, joita ei voida esittää absoluuttisen tarkasti kuvapistetekniikassa. On myös syytä muistaa, että CSS-määritykset käyttävät desimaalien erotteluun pilkun sijaan pistettä amerikkalaisen tavan mukaisesti, minkä takia myös koodiviittaukset ovat tässä muodossa. [1; 2; 21.]

3.4 Media queryt ja breakpointit

Yksi responsiivisen suunnittelun tärkeimmistä työkaluista HTML5-tekniikassa ovat media queryt. Alun perin erillistiä tulostus- ja kuvaruutunäkymiä varten suunniteltua media-määritystä laajennettiin CSS3-versiossa uusilla ominaisuuksilla, joiden avulla erilaisia näkymiä kyetään luomaan paljon laajemmalle kirjolle laitteita. [1; 2.]

Tekniikka hyödyntää yksinkertaista syntaksia, jolla pystytään paikantamaan käytetyn laitteen kuvaruudun koko joko pikseleinä tai relatiivisessa koossa. Kun määrityksiä tehdään muutama erilainen kuhunkin kokoluokkaan, saadaan sivuston peruselementit vaihtamaan muotoaan hyvinkin joustavasti. Media queryt kirjoitetaan aina muotoon

@media	not / only / and	screen / print / (max-device-width:)
Tunnistin	Looginen operaattori	Kohde, jossa looginen operaattori toimii

Media queryjen operaattorit voidaan linkittää loogisesti useisiin parametreihin, jotka saavat olla voimassa yhtäaikaisesti, jolloin sivujen ulkoasut voidaan määrittää hyvinkin tarkasti kaikissa tarvittavissa toistoympäristöissä. [1; 2.]

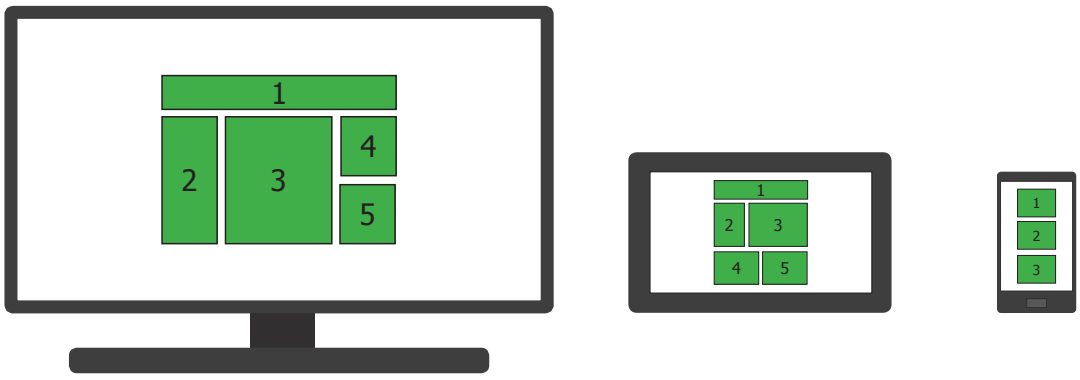
Kuvan 3 näyttökoot käsittelevät viittä sisältöelementtiä, joille on määritelty kolme relatiivista kokoluokkaa. Niistä suurimmat, eli PC-näytöt ja televisiot (eli kooltaan yli 60,5 em-yksikköä), hyödyntävät kolmipalstaisuutta, pienemmät PC-näytöt sekä kämmen-tietokoneet kahta ja mobiililaitteet vain yhtä palstaa. Koska kaikki määritykset ovat yksilöllisiä, ei elementtien keskinäistä suhdetta tarvitse ylläpitää näiden välillä, vaan kaikille voi määritellä täysin omat ulkoasunsa. [1; 2.]

Esimerkki tyypillisestä syntaksista kolmelle näyttökoolle:

```
@media only screen and
(min-width: 60em){
   Desktop-määritykset
}
```

```
@media only screen and
(min-width: 40em) and
(max-width: 60em){
   Tab-määritykset
}
```

```
@media only screen and
(max-width: 40em){
   Mobiilimääritykset
}
```



TFT-monitorin kuvaruudulla | Kämmentietokoneen kuvaruudulla | Älypuhelimen kuvaruudulla

Kuva 3: Responsiivisen sivun toimintamallin esimerkki erilaisissa näyttöko'oissa [2].

3.5 Pikselikuvat

Pikseli- eli rasterikuvat (pääasiassa JPEG-, PNG- ja GIF-formaateissa) ovat webissä vektorikuvia huomattavasti käytetympiä. Pikselikuvien toimintalogiikka perustuu siihen, että ne luodaan aina tiettyyn matriisikarttaan, jolloin alkioita kuvaavien näyttöpisteiden koko on aina toistolaitteen resoluution mukainen. Kuvia pystytään skaalaamaan CSS-määrityksillä pienemmäksi (pienellä kuvanlaadun heikkenemisellä lähinnä reuna-alueilla anti-aliasing-tekniikan puutteiden vuoksi), ja se on ollut tuettua web-suunnittelussa jo pitkään esimerkiksi prosentuaalisiin kokoihin perustuvissa sisältötageissa. [1; 2.]

Jos kuvan alkuperäinen tallennuskoko vastaa sen ympäriltä löytyvän sisältötagin kokoa, se ei oletusarvoisesti skaalaudu yli 100 %:n koon, koska pikselikuvat menettävät kuvalaatuaan, kun niitä skaalataan ylöspäin. Joskus sisältöelementit on kuitenkin suunniteltu niin, että kuvan tulisi täyttää tietty alue (esimerkiksi taustakuvan muodossa), jolloin kuvan on pakko mukautua elementin kokoon. Perinteisessä web-suunnittelussa sisältötagit suunniteltiin tiettyyn pikselipohjaiseen maksimikokoon. Tällöin pystyttiin tarkasti määrittämään, ettei tagi ylittänyt kuvan pikselidimensioita. [1.]

Täysin responsiivisessa suunnittelussa pikseleitä käytetään enää harvoin sivun sisältöelementtien koon määritykseen, ja relatiivisissa kokomääritteissä yli 100 %:n kuvakoon saaminen sisältöelementeille on mahdollista. Jos esimerkiksi alkuperäistä 400 pikselin levyistä, mutta em-koossa määritettyä kuvaa katsotaan noin 100 ppi:n tarkkuisella näytöllä, se saattaa vaikuttaa hyvälaatuiselta ja terävältä, mutta yli 300 ppi:n mobiilinäytöllä, sen kuvanlaatu heikkenee. Ongelma on siinä, että kuvalle on varattu käytettäväksi tietty tila, jonka se pyrkii täyttämään, mutta koska pikseleitä on varattu kyseiselle alueelle mobiilinäytöissä enemmän, on laitteen suurennettava kuvaa häviöllisesti yli 100 %:n koon. [2.]

CSS3:n media-attribuutti mahdollistaa useiden erillisten kuvien toiston pikselikoon perusteella, jolloin voidaan tehdä kaksi tai useampia kuvakokoja eri näyttöjen pikselitiheyksiä varten. Ongelmaksi tekniikassa muodostuu se, että useimmat korkean pikselitiheyden näytöt ovat mobiililaitteita. Lisäksi ne hyödyntävät useimmiten langatonta Internet-yhteyttä, jonka nopeus saattaa vaihdella paljonkin. Siis mitä suurempia kuvat ovat, sitä kauemmin käyttäjä joutuu lataamaan niitä, jolloin sivujen latausaika kasvaa. [2.]

3.6 Vektorikuvat

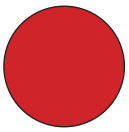
Vektorikuvat ovat kuvatyyppi, joka muodostuu geometrisia alkioita hyödyntävistä matemaattisista lausekkeista. Niiden erityispiirteisiin kuuluu rajaton tuki skaalautuvuuteen, mikä on tehnyt niistä teknisen standardin modernissa logo- ja piirroskuvasuunnittelussa sekä suurimmassa osaa fonteista. HTML-sivut tukevat tällä hetkellä (tammikuu 2014) vain SVG-kuvaformaattia, joka kirjoittaa vektorikuvan uudelleen SVG-koodin mukaisesti ryhmäksi loogisia parametreja, joiden avulla varsinainen kuva piirretään selaimeen.

SVG-grafiikka on edistyksellinen tekniikka, koska se mahdollistaa elementtien piirtämisen suoraan web-sivuille vektorimuodossa. Sen käyttöönotto on ollut kuitenkin vaikeaa, koska monet perusmääritykset pohjautuvat pikselipohjaiseen kokoluokitteluun, vaikka niiden käytöstä ollaan pyrkimässä pois.

Esimerkkejä SVG-koodin toimintalogiikasta on kuvassa 4.

Yksinkertaisin parametrein määritelty pallo:

```
<svg height="100" width="100">
<circle cx="50" cy="50" r="40" stroke="black" stroke-width="3" fill="red" />
</svg>
```



Tummansininen N-kirjain:

```
<svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink"
x="0px" y="0px" width="510.236px" height="243.78px" viewBox="0 0 510.236 243.78" enable-background="new 0 0
510.236 243.78" xml:space="preserve">
<g><g>
<path fill="#1D4458" d="M234.827,139.163c0,0.156-0.128,0.283-0.283,0.283h-
9.962c-0.156,0-0.347-0.111-0.424-0.246
l-12.207-21.428c-0.077-0.135-0.14-0.119-0.14,0.037v21.354c0,0.156-0.128,0.283-0.283,0.283h-8.418
c-0.156,0-0.283-0.128-0.283-0.283v-34.556c0-0.156,0.128-0.283,0.283-0.283h10.39c0.156,0,0.348,0.11,0.427,0.24
4l11.772,20.052
c0.079,0.134,0.144,0.117,0.144-0.039v-19.974c0-0.156,0.128-0.283,0.283-0.283h8.41
7c0.156,0,0.283,0.128,0.283,0.283
L234.827,139.163L234.827,139.163z"/>
</g></g>
</svg>
```

Kuva 4: SVG-grafiikan piirtologiikkaa [22].

Esimerkistä voi huomata, että SVG-koodilla pystyy piirtämään yksinkertaista grafiikkaa hyvin helpoilla määrityksillä. Monimutkaisemmat kuviot vaativat kuitenkin huomattavan määrän SVG-koodia, jota on hyvin vaikeaa tuottaa käsin. Koodin attribuuttitason määritykset toimivat pikselipohjaisesti, mikä tekee koodimuotoisen käytöstä haastavaa relatiivisia kokomäärityksiä hyödyntävissä sivuissa.

HTML5-tekniikan yleistyttyä myös SVG-tiedostotuki on levinnyt img-elementin ja taustakuvien yhteyteen, mikä on helpottanut web-vektorigrafiikan käyttöönottoa merkittävästi. SVG-grafiikkaa sisältäviä elementtejä pystyykin nykyään muotoilemaan täysin vapaasti hyödyntäen CSS-määrityksiä kaikilla moderneilla web-selaimilla. [26.]

Responsiivisissa sivuissa SVG-grafiikka tarjoaa tavan luoda täydellisen tarkkoja kuvakkeita ja logoja riippumatta toistolaitteen näyttötarkkuudesta. Tämän lisäksi SVG vie usein pikselikuvia vähemmän levytilaa, mikä on eduksi sivujen latausnopeudessa. Formaattia kannattaakin käyttää nykyään kaikessa grafiikassa, mikä on tarjolla vektorimuotoisena.

3.7 Responsiivinen web mobiililaitteissa

HTML5 ja responsiiviset tekniikat rinnastetaan usein visuaaliseen puoleen ja siihen, miten koko sivu käyttäytyy eri näyttöko'oissa. Tosiasiassa nykyisten mobiili- tai ”älykkäiden laitteiden” toiminta ei rajoitu pelkkään kykyyn toistaa sivut visuaalisesti, vaan laitteissa on useita erikoistoimintoja, joita hyödyntämällä käyttökokemusta näissä laitteissa voi merkittävästi parantaa. Web-tekniikkaa voi hyödyntää jo esimerkiksi seuraavilla osa-alueilla:

- puheluominaisuudet, jolla henkilö voi soittaa tai lähettää tekstiviestin suoraan sivuille upotettujen linkkien avulla (anchor tagin 'tel' ja 'sms' -protokollien kautta)
- viiteliikkeet ja kallistuksentunnistin, joilla sisältöä voi selata hiirestä poiketen erilaisin tavoin käsieleillä tai laitetta liikuttamalla
- (video)kamera, jolle kehitetään tekniikoita kohti laajennetun todellisuuden palveluita
- GPS-paikannin, jolla pystyy sivulle upotettuja valmiita karttasovelluksia selaamaan sijaintiin nähden. [2; 23.]

Tekniikka on siis vielä suhteellisen uutta, ja vielä varmasti kehittyy tulevaisuudessa. Yksi mahdollinen web-kehityksen suunta on perinteisen web-tekniikan ja mobiilisovelluksien integrointi ”jokapaikan tietotekniikkaan” (Ubiquitous Computing) tavalla, jolla sivuista voisi muotoutua interaktiivisia palveluympäristöjä. [23.]

Valtava kirjo laitteita kytketään jo nykyään verkkoyhteyteen, ja erilaisten mobiililaitteiden tekniikan kehitys on yksi tekniikan tehokkaimmista kasvuympäristöistä. Koska harvoja sivuja tai sovelluksia kannattaa enää tehdä staattisiksi, on responsiivisen ulkoasusuunnittelun tärkeys varmasti vain kasvamassa tulevaisuudessa. [23.]

4. Insinöörityön tutkimuskohteet

Insinöörityössä tutkittiin kolme erillistä web-sivustoprojektia, joissa hyödynnettiin responsiivisia tekniikoita. Sivuista kaksi toteutettiin Joomla-sisällönhallinta-järjestelmään, ja ne on tilaajan pyynnöstä poistettu julkisesta opinnäytetyöstä (luvut 4.1 ja 4.2). Kolmas projekti toteutettiin modulaarisena PHP-sivustona, joka on osana julkista työtä.

4.3 Netrakin kotisivu (modulaarinen PHP)

Asiakasprojekti, jossa pääsin tekemään responsiivista sivustoa täysin itsenäisesti oli mainostoimisto Indie Group Oy Helsingin asiakkaalle Netrak Oy:lle tehtävä yrityssivusto. Projektiin kuului seitsemän erillistä sivua, joista kaikille Indie Group oli suunnitellut alustavan Photoshop-ulkoasun hyödyntäen näkymää enimmäisleveydelle. Sivun visuaalinen osuus oli korkealla prioriteetilla ja sen ohessa omaksi tehtäväkseni muodostui myös responsiivisuuden toiminnallinen suunnittelu.

Koska sivustomäärä oli suppea eikä asiakas itse pyrkinyt päivittämään sivuston sisältöä tai päivittämään mitään uutis-, blogi- tai muuta aktiivista rakenneosasta, valittiin tekniikaksi räätälöidysti rakennettava HTML5-sivusto valmiiden CMS-tekniikoiden sijaan. Asiakkaalle oli myös tärkeää, että sivusto toistuisi mahdollisimman monipuolisesti kaikilla toistolaitteilla, joten Flash-tekniikkaa päätettiin välttää erikoisratkaisuissa, kuten animoiduissa bannereissa tai galleriassa.

Näiden määrityksien mukaan parhaaksi toteutustekniikaksi valittiin PHP- ja HTML5-perusrakenne, jonka ohella hyödynnettiin Javascript-pohjaisia kuvanvaihtajia. Ne olivat käyttöönoton hetkellä tuettuja jo lähes kaikissa ajanmukaisissa laitteissa ja niiden selaimissa.

Tuki HTML4-tekniikkaan

Sivustoprojekti sijoittui sellaiseen ajankohtaan, jossa HTML5-tekniikka oli jo yleisesti käytössä, mutta jolloin osa käyttäjistä hyödynsi edelleen web-selaimia, joista puuttui tuki HTML5-tekniikkaan. Tämän vuoksi pohdittiin myös, oliko mahdollista tehdä sivustoa tukemaan HTML4- tai aiempia versioita tukeva vaihtoehtoisrakenne.

Koska projektin budjetti oli kuitenkin rajattu ja erilaisten tekniikoiden integroiminen breaking pointeja järkevästi hyödyntävästi osoittautui ajan puolesta raskaaksi toteuttaa, päätettiin siitä luopua. Päätökseen olla tekemättä vanhojen selainversioiden kanssa yhteensopivaa sivustoa vaikutti myös se, että suomalaisilla kävijöillä tekniikkaa tukemattomia web-selaimia oli oletettu olevan käytössä enää suhteellisen vähän. Vertailuarvoksi otettiin Iltasanomien vuoden 2012 marraskuussa käyttäjien selainversioista julkaisema tutkimus, jonka oletettiin kuvaavan kohtalaisesti yrityksen asiakkaita. [24.]

Sivuston kuvat

Kuvamateriaalia oli suunniteltu jokaiselle sivulle, ja se suunniteltiin käytettäväksi staattisissa em-pohjaisissa sisältöruuduissa. Em-koot laskettiin muun sisällön mukaan suoraan alkuperäisen kuvan resoluutiosta, joka jaettiin lukuarvolla 16, eli 1 em:n oletuskoolla. Korkeatarkkuusnäyttöjä ei otettu tässä sivustossa erikseen huomioon, vaan kaikki kuvamateriaali tuotettiin oletusarvoisesti 16 px/em -koolle. Sivuston kaikki tavallinen kuvamateriaali noudatti kahta kokostandardia:

Matala kuva oli yleisimmin käytetty kuvaformaatti, joka aseteltiin sisältöpalstan oikealle puolelle ja sen alle. Matalat kuvat oli mitoitettu yhden palstan levyiseksi sekä suhteessa 2:3-standardiformaattiin, ja ne saattoivat sisältää sivuston sisäisiä linkkejä.

Korkea kuva oli työpöytäkoossa ylin tekstin oikealta puolelta löytyvä kuvapohja. Sen leveys oli määritelty myös yhteen palstaan, mutta korkeus oli kahden matalan kuvan ja niiden välisen marginaalin korkeuksien summa. Tällä tavoin saatiin kämmentietokoneille optimoidusta näkymästä tasakorkuinen, kun kaksi matalaa kuvaa asettui päällekkäin korkean kuvan oikealle puolelle. Muutamissa sivuissa korkean kuvan paikkaa käytettiin animoidun kuvakokoelman tekoon, jolloin useampaa kuvaa vaihdettiin tietyin aikavälein.

Matala kuva
(12,5 x 18,75 em / 300 x 200 px)

Korkea kuva
(26 x 18,75 em / 300 x 416 px)

Kuva 16: Matala ja korkea mallikuva [25].

Sivutyypit

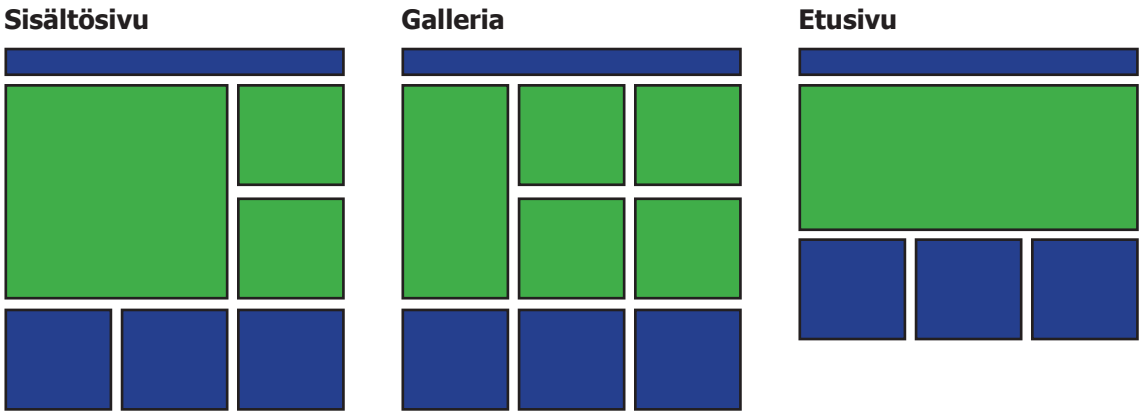
Rakenteellisia sivumalleja Netrak-projektissa oli muutama erilainen. Tärkeinä kriteereinä määriteltiin muuttuvan sisällön asemoituminen työpöytä- ja kämmentietokoneitten näytöillä. Sivutyypit luokiteltiin seuraavasti:

Sisältösivu edusti sisältörakenteeltaan 2 + 1 -palstajakoa, eli varsinaiselle muuttuvalle tekstisisällölle oli varattu kahden palstan tila ja oikealle asiaan liittyviä kuvia. Koska sisällön pituus ei ole vakio, oli myös kuvien lukumäärän vaihdeltava sisällön suhteellisen pituuden mukaan. Kaikista lyhyimpiin sivuihin suunniteltiin oikealle vain yksi korkea kuva ja muihin osioihin lisäksi 1—2 matalaa kuvaa.

Galleria eli referenssit-sivu oli yksittäinen sivusto, joka ei toteuttanut sisältösivun rakennemääritelmää ja joka piti siksi käsitellä omana kokonaisuutenaan. Galleriasivun kaikki sisältöosiot (teksti- ja kuvaosiot) olivat yhden palstan levyisiä, mutta korkeudeltaan joko yhden matalan tai korkean kuvan kokoisia.

Etusivu esiintyi referenssit-sivun tapaan vain yhtenä sivuna, mutta on rakenteellisesti sisältösivusta merkittävästi poikkeava. Etusivun merkittävä ero on sen bannerissa, jonka responsiivisuus toimii aivan omalla tavallaan.

Tämän luokittelun perusteella kaikkien sivujen rakenteet pystyttiin yksinkertaistamaan niin, että ne noudattivat jotain näistä kolmesta sivurakenteesta. Kun ensimmäinen sisältösivumalli oli luotu, sitä pystyi monistamaan kaikkiin samankaltaisiin sivuihin, mikä helpotti työtä merkittävästi. Kuvassa 17 sivujen responsiivisuus kuvataan kaaviolla.

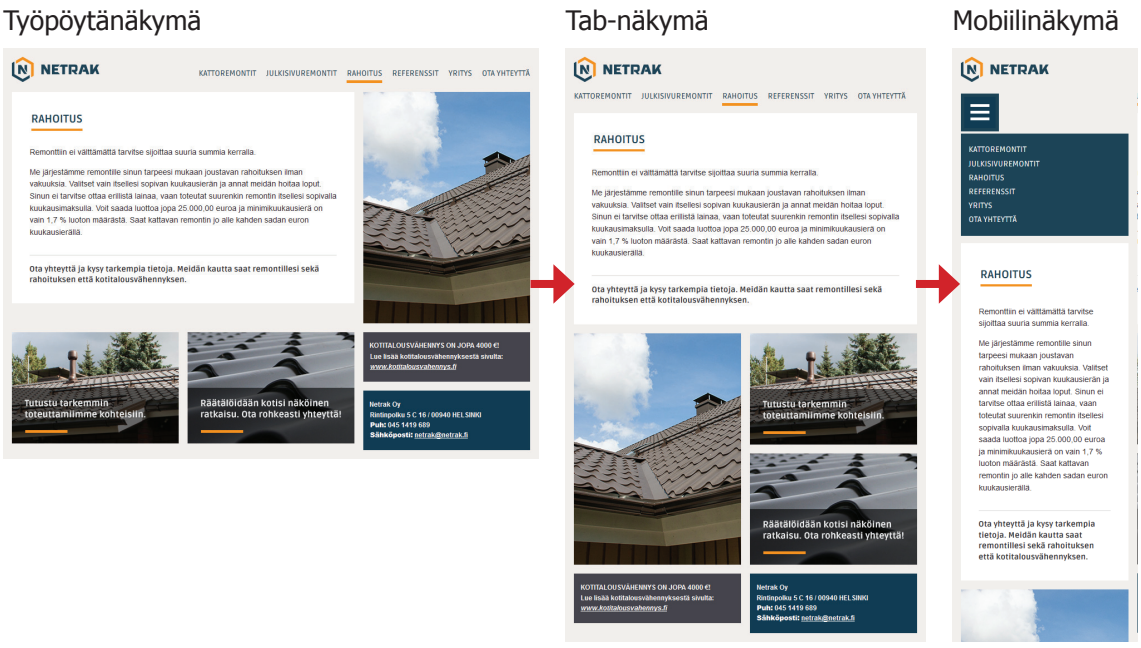


Kuva 17: Netrakin oletussivumallit [25].

Responsiivisen ulkoasun suunnittelu

Indie Groupin toimittaman ulkoasumallin perusrakenne oli jaoteltu logo- ja navigaatioriviin ja kolmeen yhtä leveään palstaan, jota oli tarkoitus hyödyntää työpöytänäkymässä. Indie Group ei ollut itse suunnitellut valmiiksi, miten responsiivisuuden tulisi toteutua, joten tämä osuus mietittiin yhdessä valitun tekniikan pohjalta.

Useimmissa sivuissa ulkoasun rakenne oli jaoteltu sisältöosuuteen ja oikean puolen kuviin, joiden alapuolella oli vielä asiaan liittyviä kuvia sekä tekstiä sisältäviä osioita. Painoarvoltaan korkein oli tekstiosuus, joka vei vähintään 2/3 käytössä olevasta leveydestä, ja muut osiot olivat aina saman levyisiä. Yhdenmukaiset kolme palstaa loivat mainion pohjan suunnitella myös varsinaiset breaking pointit kolmeen osaan, eli ”työpöytä-”, ”kämmentietokone-” ja ”mobiilinäkymään”, jolloin siirryttäessä pienempään ruutukokoon vähennettiin palstamäärää aina yhdellä (minimissään yksipalstaiseen malliin asti). Kuvassa 18 esitellään sivuston yhden sisältösivun responsiivisuus.



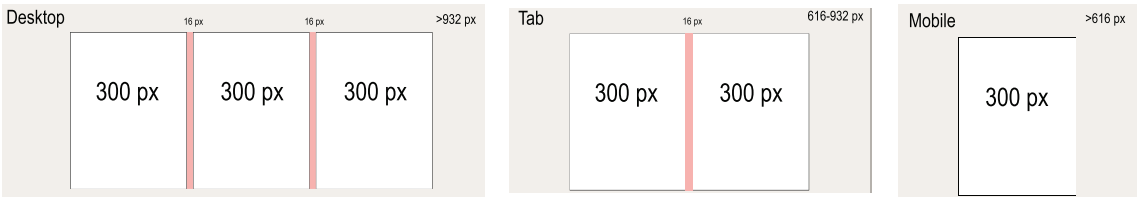
Kuva 18: Netrak-sivuston responsiivisuus [25].

Vaikka perusrakenne oli suunniteltu suhteellisen yksinkertaiseksi, oli responsiivisuus mietittävä hieman tarkemmin, koska sisällöt oli tarkoitus erotella toisistaan leveähköllä marginaalilla. Marginaalin käyttö ei ollut kuitenkaan aivan itsestäänselvää, koska oikean-

puolimmaiseen sisältöosuuteen ulkomarginaalia ei haluttu luoda, jotta nämä sisältöalueet menisivät aina oikean reunaan mukaan tasan. Kun erilaisia näkymiä oli kolme, oli jokaiselle laatikolle mietittävä erillinen toimintalogiikka, joka toimisi kolmessa eri ympäristössä ja asettuisi aina oikeaan kohtaan. Kaikista haastavimmaksi osioksi muodostui referenssit-sivusto, jossa korkeiden ja matalien kuvien järjestys vaihteli jokaisessa kuvassa.

Prosentuaaliset kokomääritykset eivät olleet tässä projektissa merkittäviä, koska sivu oli suunniteltu toimimaan niin, että sisältöalueet pysyisivät mahdollisimman samankokoisina eikä pienellä skaalauspotentiaalilla ollut niin suurta merkitystä luettavuuden kannalta. Kaikki sisältölaatikot toimivat siis lukituissa kokoluokissa, ja väliformaatteja varten sivun ulkoinen reuna-alue joko venyi tai kapeni breaking pointien välillä.

Koska sivujen oli toimittava helppolukuisesti kaikilla näyttötarkkuuksilla, päätettiin mittakooksi luoda ’em’-pohjainen kokomääritys. Yhden palstan leveydeksi valittiin 300 pikseliä, mikä saatiin käännettyä em-mittakaavaan jakamalla se 1 em:n oletuskoolla 16, jolloin palstaleveydeksi saatiin 18,75 em. Lisäksi ulkoasussa oli tarpeen huomioida kerrallaan suurimmillaan kahdelle välimarginaalille varattava tila, jolle otettiin suhteessa sopivan kokoisen välin luova 1 em. Koko sivuston laajimmaksi sisältöosuuden leveydeksi saatiin siis 58,25 em (932 px). Kuva 19 esittelee palstojen teknisen rakenteen näkymissä.



Kuva 19: Palstoitusmalli [25].

Vaikka sivuston perusrungon koot oli saatu näin laskettua, varsinaiset breaking pointit oli mietittävä hiukan eri kokoon, jotta sivu toimisi myös raja-alueen kokoisilla näytöillä hyvin. Kaksipalstaisen ulkoasun minimileveys oli 38,50 em (18,75 em * 2 + 1 em), mutta kun molemmille reunoille laskettiin minimissään 1 em:n reuna, saatiin ensimmäiseksi breaking pointiksi 40,5 em. Tätä pienemmillä näytöillä mobiilinäkymä otettiin automaattisesti käyttöön. Kolmepalstaisessa näkymässä (18,75 em * 3 + 2 em) minimileveys oli 58,25 em, johon lisäämällä rajat saatiin 60,25 em.

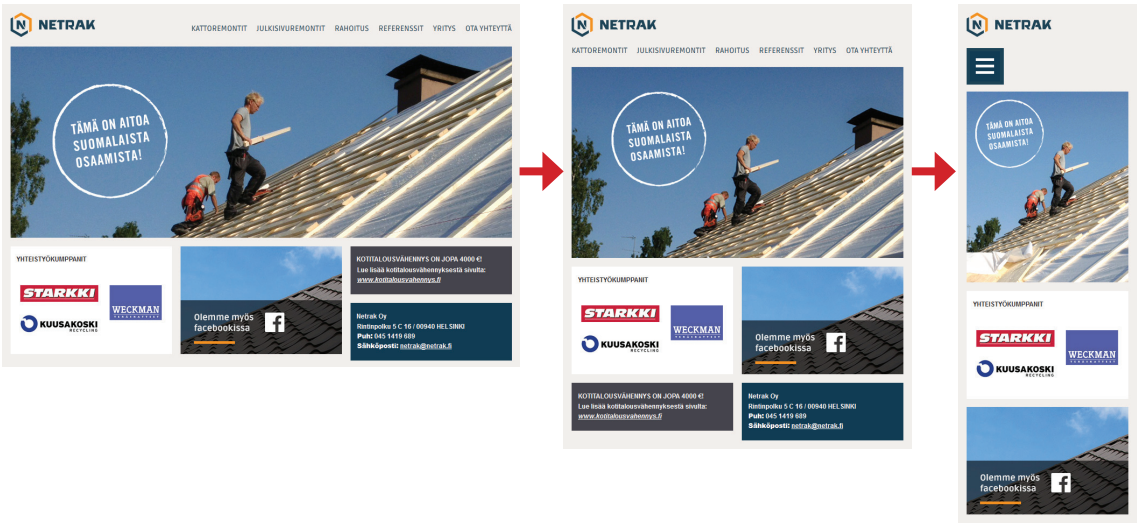
Koska kaikki muut kokoluokat olivat yhden desimaalin mittaisia, pyöristettiin viimeiseksi breaking pointiksi 60,5 em. Sivuston lopullisiksi kokoluokiksi saatiin siis: **Mobiili**: <40,5 em, **Tab**: 40,5-60,5 em, **Desktop**: >60,5 em.

Etusivubannerin responsiivisuus

Peruskuvien lisäksi etusivulle tuotettiin erityinen animoitu kuvabanneri, jolla oli erityisvaatimus olla kolmen palstan levyinen ja suurimmassa koossaan 22,1875 em:n (355 px:n) korkuinen, eli vähän yli 2,5 kertaa korkeuttaan leveämpi. Koska myös etusivulle tarvittiin responsiivinen toiminta, oli etusivubannerin toiminta mietittävä erityisen tarkkaan.

Haasteeksi muodostui se, että mikäli bannerissa olisi hyödynnetty vain tämän mitta-suhteen mukaisia kuvia, olivat kuvat varsinkin mobiilitasolla kutistuneet todella pieniksi, jolloin kuvien sisältämä teksti olisi muuttunut lähes lukukelvottomaksi. Siksi etusivun kuva-animaatiolle päätettiin jokaiselle kokoluokalle luoda erilliset kuvat, joiden korkeus pysyi samana, mutta kuvien (Photoshopissa tehty) rajaus muuttui kuvasuhteen myötä.

Kuvassa 20 esitellään, miten etusivubanneri mukautui kuhunkin näkymään.

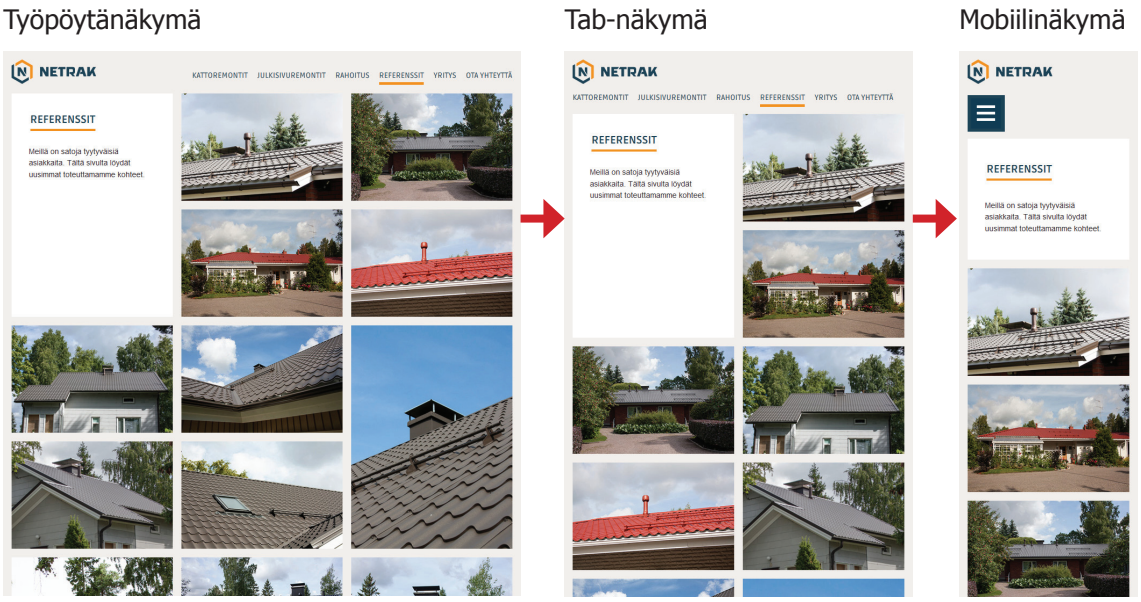


Kuva 20: Etusivubannerin responsiivisuus [25].

Banneri toteutettiin Javascript- ja PHP-tekniikoilla, jotta se saatiin toimimaan mahdollisimman kattavasti riippumatta toistolaitteesta. Kuvia tehtiin neljä jokaiseen näkymään, ja niitä vaihdettiin yksi kerrallaan noin viiden sekunnin välein. Samaa toimintalogiikkaa hyödynnettiin myös muiden sivujen (katto- ja julkisivuremontit -kohtien) animoiduissa kuvissa, jotka muualla sijoittuvat sisällön oikealle puolelle ja aina vain yhden palstan levyisinä.

Referenssit-sivuston responsiivisuus

Referenssit-sivun galleria oli teknisesti yksi sivun haastavimmista osioista. Sivun sisältö ei noudattanut tavallista rakennelogiikkaa, ja kuvien korkeudet vaihtelivat matalista korkeisiin ja vaihtelevassa järjestyksessä. Sivukokonaisuus ja sen elementit piti siksi suunnitella erikseen kaikille kolmelle kokomäritykselle. Kuva 21 esittelee, miten gallerian sisältö mukautuu eri näkyviin.



Kuva 21: Referenssit-sivun responsiivisuus [25].

Ongelmia tuotti eritoten se, että oikeanpuolimmaisen sarakkeen kuville ei saanut missään näkymässä tulla oikeaa ulkomarginaalia, mutta kuvien järjestys ja rivitys vaihtui täysin siirryttäessä desktop-näkymästä tab-näkymään. Lisäksi koska kuvien korkeus vaihdeli, piti kuvista luoda pareja ympäröimällä ne erillisillä div-elementeillä jotka vaihtuivat pystyformaatista vaakaan juuri toivotusti. Hyödyntämällä monitasoista

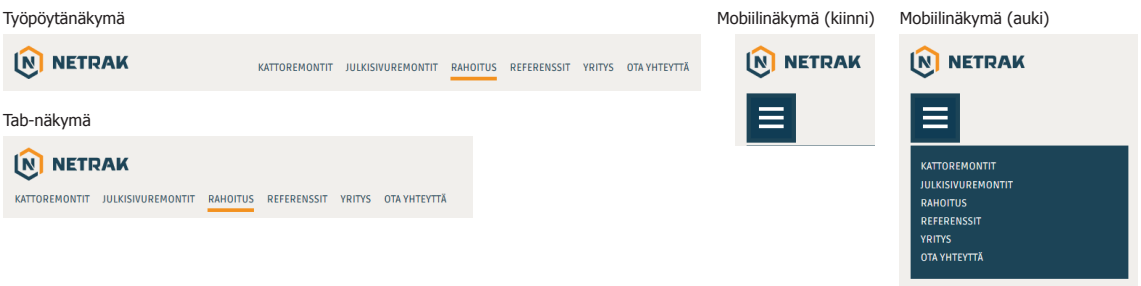
class- ja ID-rakennetta kuvissa ja niitä ympäröivissä div-elementeissä pystyttiin responsiivisuus tuottamaan täsmällen halutulla tavalla.

Sivun kuville luotiin vielä erillinen toiminto, jolla tab- ja työpöytä-näkymien kuvat sai suurennettua niitä klikkaamalla. Toiminto päätettiin toteutettaa Javascriptillä, jotta se tukisi selaimia mahdollisimman laajasti. Mobiilikoossa toiminto piti erikseen estää, koska näytön leveys riitti juuri ja juuri kuvien toistamiseen pystysuunnassa, jolloin kuvien klikkaus esimerkiksi vahingossa kosketusnäytöllä voisi johtaa käyttöongelmiin.

Navigaation responsiivisuus

Indie Group oli määrittänyt sivun navigaation alun perin vain päänäkymään tavallisena tekstinä, joka sijoitettiin sivun oikeaan laitaan logon viereen. Aktiivista valikkokohtaa korostamaan sen alle luotiin paksunnettu alleviivaus. Tab-kokoisen sivun leveydessä navigaatiopalkki ei enää kuitenkaan mahtunut logon viereen, vaan se piti siirtää logon alle, mutta muuten ulkoasultaan samanlaisena.

Mobiilinäkymässä tilaa ei ollut enää senkään vertaa, ja navigaatiota varten täytyi tehdä erillinen toiminto, jossa valikko piilotettiin oletusarvoisesti ja klikkaamalla suurta navigaatiokuvaketta sai valikon avattua tai suljettua. Mobiilinäkymälle ei enää rakennettu erillistä aktiivitilan tunnistinta, mutta hover-tilaa varten määriteltiin sille korostettu oranssi tekstiväri. Kuva 22 esittää, miten logo ja navigaatio mukautuivat eri näkymiin.



Kuva 22: Netrak-navigaation responsiivisuus [25].

5 Yhteenveto

Insinöörityössä perehdyttiin tapoihin, joilla responsiivista web-suunnittelua voidaan tehdä HTML5-tekniikoiden avulla. Työssä tutkittiin kolmea yrityssivustoa, niiden rakennetta sekä tapoja, joilla sivustojen eri tekniset osat toteutettiin huomioiden laiteriippumattomuuden seikat.

Teknisten web-standardien pitkää kehityshistoriaa seurattaessa on selvää, etteivät responsiiviset tekniikat ole saavuttaneet vielä teknistä huippuaan vaan kehitys jatkuu vielä pitkään esimerkiksi kuvien, videotoiston ja muiden osa-alueiden parissa. Tämä kuitenkin on olemassa olevien metodien kehitystyötä, ja kun huomioidaan esimerkiksi HTML4:n pitkä elinkaari, voidaan olettaa HTML5:n olevan nykymuodossaan tuettu vielä pitkään. Tällöin nykyiset tekniikat eivät todennäköisesti katoa mihinkään pitkään aikaan.

Työstä selvisi, että responsiivista eli mukautuvaa web-suunnittelua voi nykymuodossaan hyödyntää jo hyvinkin monenlaisissa sivustoissa yksinkertaisista muutaman sivun toteutuksista suuriin järjestelmiin, joissa sisältö ja rakenne ovat eristyksissä toisistaan. Hyödyntämällä standardin mukaisia toimintatapoja varmistetaan sivun toimivuus mahdollisimman monessa kohdelaitteessa ja -selaimessa, joiden määrä on jatkuvassa kasvussa. Vaikka vanhoilla staattisilla sivuilla on vielä periaatteessa laajin selaintuki, on niiden tekninen rajoittuneisuus käytettävyysrajoite, minkä takia ei uusia sivuja enää kannata suunnitella staattisiksi.

Yleisellä tasolla web-suunnittelun määritelmä saattaa tulevina vuosina siirtyä nimenomaan kohti responsiivista suunnittelua, kun staattiset sivut eivät enää palvele asiakkaita riittävän laajasti. Tämän ovat tiedostaneet monet sisällönhallintajärjestelmien kanssa toimivat tahot, jotka joutuvat huomioimaan sisällön joustavuuden sivusto-rakenteissaan.

Responsiivisessa suunnittelussa ei siis ole kyse niinkään uuden vaihtoehtoisen tekniikan luomisesta vanhan rinnalle vaan siirtymisvaiheesta koko web-tekniikan uudelle kehitysasteelle, jossa laitteet eivät enää rajoita suunnittelua vaan mukautuvat siihen.

Lähteet

1. Frain, Ben. 2012. Responsive web Design with HTML5 and CSS3. Packt Publishing.

2. Williamson, James. 2010. HTML5 First Look. Verkkodokumentti. <http://www.lynda.com/HTML-5-tutorials/html5-first-look/67161-2.html>. Luettu 26.3.2014.

3. Williamson, James. 2012. Responsive Design Fundamentals. Verkkodokumentti. <http://www.lynda.com/web-Responsive-Design-tutorials/Responsive-Design-Fundamentals/104969-2.html>. Luettu 2.8.2013.

4. Browser Display Statistics. Verkkodokumentti. w3schools. <http://www.w3schools.com/browsers/browsers_display.asp>. Luettu 26.3.2014.

5. Patterson, Steve. The 72 PPI Web Resolution Myth. Verkkodokumentti. <http://www.photoshopessentials.com/essentials/the-72-ppi-web-resolution-myth>. Luettu 6.2014.

6. Kopp, Carlo. 2005. Thin Film Transistor Liquid Crystal Display Technology. Verkkodokumentti. <http://www.ausairpower.net/OSR-0398.html>. Luettu 26.3.2014.

7. List of Displays by Pixel Density. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/List_of_displays_by_pixel_density>. Luettu 26.3.2014.

8. Apple iPhone. Verkkodokumentti. gsmarena. <http://www.gsmarena.com/apple_iphone-1827.php>. Luettu 26.3.2014.

9. A history of HTML. 1998. Verkkodokumentti. w3.org. <http://www.w3.org/People/Raggett/book4/ch02.html>. Luettu 26.3.2014.

10. Introduction to HTML4. Verkkodokumentti. w3.org <http://www.w3.org/TR/html4/intro/intro.html>. Luettu 26.3.2014.

11. Arandilla, Rachel. Web Design History: From The Beginning. Verkkodokumentti. <http://www.1stwebdesigner.com/design/web-design-history-from-the-beginning/>. Luettu 26.3.2014.

12. West, Angela. 2010. 20 years of Adobe Photoshop. Verkkodokumentti. <http://www.webdesignerdepot.com/2010/02/20-years-of-adobe-photoshop>. Luettu 26.3.2014.

13. Nielsen, Jakob & Loranger, Hoa. 2006. Prioritizing Web Usability. New Riders.

14. Vilches, Jose. 2011. Adobe stops mobile flash development, will focus on HTML5. Verkkodokumentti. <http://www.techspot.com/news/46192-adobe-stops-mobile-flash-development-will-focus-on-html5.html>. Luettu 18.3.2014.

15. Jebaraj, Daniel. 2011. HTML5 or Silverlight? Verkkodokumentti. <http://www.infoq.com/articles/Html5-or-Silverlight>. Luettu 26.3.2014.

16. Park, Anthony & Watson, Mark. 2013. HTML5 video at Netflix. Verkkodokumentti. <http://techblog.netflix.com/2013/04/html5-video-at-netflix.html>. Luettu 18.3.2014.

17. Pieters, Simon. 2013. Differences from HTML4. Verkkodokumentti. <http://www.w3.org/TR/html5-diff/>. Luettu 26.3.2014.

18. Knight, Kayla. 2009. Fixed vs. Fluid vs. Elastic Layout: What's The Right One For You? Verkkodokumentti. <http://coding.smashingmagazine.com/2009/06/02/fixed-vs-fluid-vs-elastic-layout-whats-the-right-one-for-you>. Luettu 26.3.2014.

19. Hernandez, Guil. 2013. Thinking Ahead: CSS Device Adaptation With @viewport. Verkkodokumentti. <http://blog.teamtreehouse.com/thinking-ahead-css-device-adaptation-with-viewport>. Luettu 17.2.2014.

20. Plafke, James. 2013. Retina redefined: LG makes world's thinnest, highest pixel density smartphone LCD ever. Verkkodokumentti. <http://www.extremetech.com/mobile/164415-retina-redefined-lg-makes-worlds-thinnest-highest-pixel-density-smartphone-lcd-ever>. Luettu 24.1.2014.

21. Schaeffer, Kyle. 2008. CSS Font-Size: em vs. px vs. pt vs. percent. Verkkodokumentti. <http://kyleschaeffer.com/development/css-font-size-em-vs-px-vs-pt-vs>. Luettu 20.3.2014.

22. Try it yourself. Verkkodokumentti. w3schools.com. <http://www.w3schools.com/svg/tryit.asp?filename=trysvg_circle>. Luettu 26.3.2014.

23. Smart Device. Verkkodokumentti. Wikipedia. <http://en.wikipedia.org/wiki/Smart_device>. Luettu 26.3.2014.

24. Suosituimmat Internet-selaimet Suomessa marraskuussa 2012. 2012. Verkkodokumentti. Ilta-sanomat. <http://blogit.iltasanomat.fi/kehitys/2012/12/18/suosituimmat-internet-selaimet-suomessa-marraskuussa-2012/>. Luettu 6.2.2014.

25. Netrakin kotisivut. 2013. Verkkodokumentti. netrak.fi.
<http://www.netrak.fi>. Luettu 26.3.2014.

26. Can I use SVG in HTML img element?. Verkkodokumentti. caniuse.com.
<http://caniuse.com/svg-img>. Luettu 2.5.2014